INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION

INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
34100 TRIESTE (ITALY) - P.O.B. 586 - MIRAMARE - STRADA COSTIERA 11 - TELEPHONE: 2240-1
CABLE: CENTRATOM - TELEX 460392-I

SECOND SCHOOL ON ADVANCED TECHNIQUES
IN COMPUTATIONAL PHYSICS
(18 January - 12 February 1988)

SMR.282/4

AN INTRODUCTION TO FORTRAN 77

M.METCALF

CERN, DD Division, Geneva, Switzerland

CERN
COMPUTER
CENTRE

DD/US/11
July 12, 1982

Revised June 1986

A N I N T R O D U C T I O N T O F O R T R A N 7 7

--=0=--

Mike Metcalf

This writeup is a copy of the transparencies used for the lectures on FORTRAN 77 given by Mike Metcalf in June 1981 and again in June 1982.

In the text BOLD type indicates features which are new or have changed with FORTRAN 77.

LANGUAGE ELEMENTS

CHARACTER SET = + - * / ( ) , . $ ' :

A-Z 0-9 blank

Collating sequence A<B<C.....<Z

0<1<2.....<9

not specified whether 0 is less than or greater than A,

but blank is less than both 0 and A.

SOURCE FORM

label	cont.	code	free
1-5	6	7 - 72	73-80

1-19 continuation lines, must be blank in cols. 1-5

COMMENT LINES

	1	2 - 72	73-80
either	C		
or	*		<-- preferred
or			anything

Can appear anywhere, even between continuation lines

SYMBOLIC NAMES

1-6 characters (1st. a letter) for subroutine and variable names, etc.

CONSTANTS

Integer	-1000	99	
Real	1.	-1.E-3	2E4
Double Precision	1.D1	-1.D-3	2D4
Complex	(1,.99E-2)		
Logical	.TRUE.	.FALSE.	
Character	'ANYTHING'		

VARIABLES

Explicitly or implicitly typed variables of the same types as the constants:

INTEGER I	I-N by default
REAL A	A-H,O-Z by default
DOUBLE PRECISION PRECIS	
COMPLEX CURRNT	
LOGICAL PRAVDA	
CHARACTER LETTER	length 1 by default

ARRAYS

Definition:

A(10)

VECTOR(-10:5) — upper and lower bounds of a dimension

MESH(-10:5,-20:-1,0:1,-1:0,2,2,2) — up to 7 dimensions

Reference:

A(integer expression)

A(I*J+MOD(M,4)**3+MARR(2))

function (MOD) — array reference (MARR(2))

CHARACTER VARIABLES, ARRAYS AND SUBSTRINGS

CHARACTER*4 A, B(6), C(9)*10

default length (*4) — specified length (*10)

(if default length not given, value 1 assumed)

A= 'WORD'

--> A(2:3) has value 'OR'

default = 1 — default = length

so A(:) has value 'WORD'

B(3) = 'LOOK'

B(3)(3:3) has value 'O'

B(3)(:2) has value 'LO'

Problems:- no mapping defined, so cannot mix with other types in COMMON or EQUIVALENCE

- no null string
- no variable length variables

SPECIFICATION STATEMENTS

TYPE STATEMENTS

INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL, CHARACTER

can be combined with IMPLICIT:

```
IMPLICIT REAL (A-B, D-K, M-Z), LOGICAL(L), CHARACTER*4(C)
```

note length

If you like 'strong typing' use

```
IMPLICIT LOGICAL (A-Z)
REAL list
INTEGER list
  etc.
```

PARAMETER AND DIMENSION

symbolic name — integer constant expression

```
PARAMETER (LENGTH=9)
DIMENSION A (LENGTH),B(4*LENGTH),C(LENGTH+20/LENGTH)
```

symbolic constant

length not specified — constant expressions

```
CHARACTER FORM*(*)
PARAMETER (LSQ=LENGTH**2,PI=355./113.,FORM='(20I9)')
DIMENSION D(LSQ)
```

specifies length

```
11  PI=22./7.   (-- compiler error
    AREA= PI*RAD**2
    DO 1 I=LENGTH,LSQ,LENGTH
      .
      .
1 CONTINUE
```


COMMON

blank: COMMON // list or COMMON list

named: COMMON /name/list

or even: COMMON/n1/l1/n2/l2/n1/l3

UGH! — cumulative

Combine with PARAMETER:

```
PARAMETER (NTRACK=20)
COMMON/TRACKS/XPOINT(NTRACK),YPOINT(NTRACK)
```

(propagating through a whole program with PATCHY, HISTORIAN or UPDATE)

EQUIVALENCE

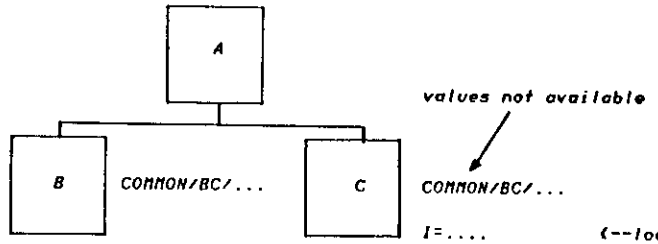
```
EQUIVALENCE (list) [,(list)]...
```

at least two names (not appearing as dummy arguments)

```
DIMENSION A(10,10)
EQUIVALENCE (A,PULSE)
```

1st. element referenced by array name

SAVE



values not available

I=.... (--local variable

undefined after return unless you specify

```
SAVE /BC/,I
```

BUT dummy statement on CDC and IBM!

EXTERNAL

```
      SUBROUTINE  ARSR
      EXTERNAL SQRT
      .
      .
      Y=SQRT(X)
      .          <-- user supplied function rather than intrinsic
      END
      FUNCTION  SQRT(ARS)
           etc.
```

INTRINSIC

```
      SUBROUTINE  DC
      INTRINSIC  SQRT
      .          <-- pass certain INTRINSIC functions in arguments
      .              (details later)
      CALL  SUBA(X,Y,SQRT)
      .
      .
      END
      SUBROUTINE  SUBA(A,B,FUNC)
      B=FUNC(A)
      END
```

DATA Can appear **anywhere** in program unit in code

```
      DATA  nlist/clist/,nlist/clist/
```

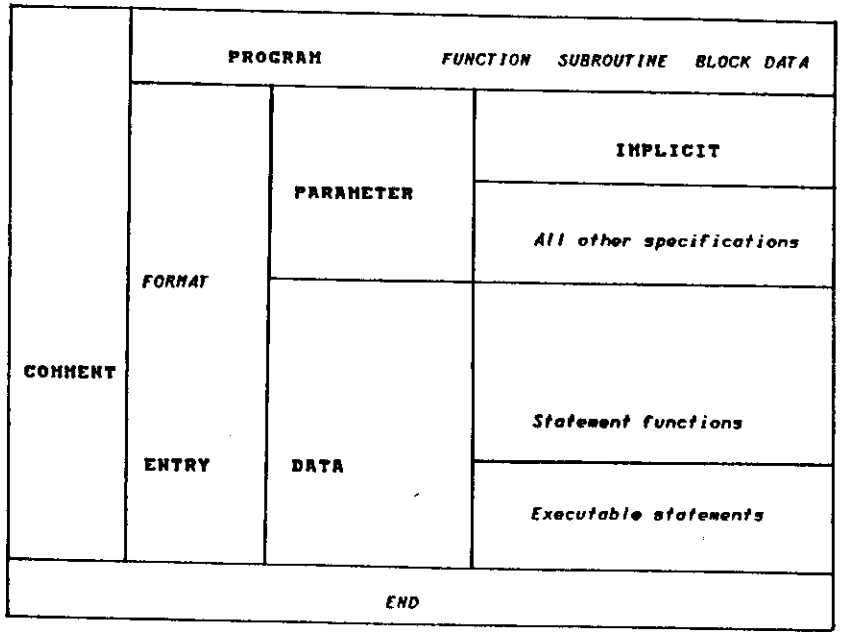
integer	=	integer
real	=	real
double precision	=	double precision
complex	=	complex
logical	=	logical
character	=	character

eg.

```
      CHARACTER*4  STR
      DATA  A/1/,I/3.1/,STR(2:3)/'AB'/
                 <-- I has value 3
      PARAMETER (L=5, ONE=1.0)
      DATA ((X(J,I),I=1,J),J=1,L,2)/9*ONE/
```

```
          X . . . .
          . . . . .
      J   X X X . .
          . . . . .   Implied Do-loop
          X X X X X
              -->
               I
```

ORDER OF STATEMENTS



EXPRESSIONS AND ASSIGNMENT STATEMENTS

ARITHMETIC EXPRESSIONS

**		order of precedence
*	/	
+	-	of operators

Expressions are evaluated left-to-right

A+B+C --> (A+B)+C

except for exponentiation which is right-to-left

A**B**C --> A**(B**C)

Manuals contain tables defining type of result of combinations of various types of operand e.g.

A*I is type REAL

and similarly for assignments v=e e.g.

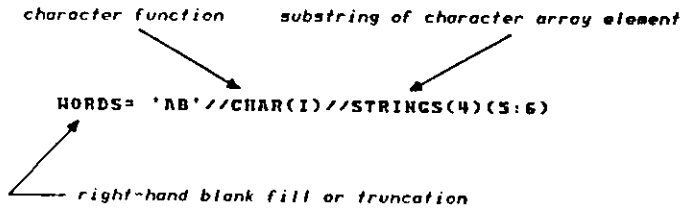
I=A

CHARACTER EXPRESSIONS

New operator // stands for concatenation e.g.

'AB'//'CD'//'EF' has value 'ABCDEF'

Character assignment:



RELATIONAL EXPRESSIONS

Operators .LT. .LE. .EQ. .NE. .GT. .GE.

IF(A.NE.B)...

Mixed types allowed

IF(A+B.NE.I-J)... ---> IF(((A+B)-(I-J)).NE.0.)...

LOGICAL EXPRESSIONS

Operators	.NOT.	order of precedence
	.AND.	
	.OR.	
	.EQV. .NEQV.	

LOGICAL I,J,K,L

expression .NOT.J.AND.K (value .TRUE. or .FALSE.)

assignment L=(.NOT.K.AND.J. NEQV. .NOT.L).OR.K

FLOW CONTROL STATEMENTS

GO TO

Unconditional GO TO 6

Computed GOTO (6,10,20) I(K)**2 ← *integer expression*

* TRANSFER TO NEXT STATEMENT IF <1 OR >3
* comma optional after closing bracket

Assigned ASSIGN 6 TO K <-- *will meet in FORMATS*

GOTO K [(4,6,10)] ← *safer to specify optional list*

IF

Arithmetic IF(16.0*P) 1, 2, 3

Logical IF(16.0*P.LT.0.) GOTO 1

BLOCK a)

```
IF(exp) THEN
     statements          <-- executed if exp .TRUE.
ENDIF
```

b)

```
IF(exp) THEN
     statements          <-- if .TRUE.
     ELSE
     statements          <-- if .FALSE.
ENDIF
```

c)

```
IF(exp1) THEN
     statements
     ELSEIF(exp2) THEN
     statements
     ELSEIF(exp3) THEN
     statements
ENDIF
```


Can nest IF-blocks like DO-loops, but not across DO-loop boundaries

```
IF(exp1) THEN                                  }
     IF(exp2) THEN           }                 }
          statements         }  level 2        }
          ELSE               }                 }
          statements         }                 }
     ENDIF                                     }
     ELSEIF(exp3) THEN                         }   level 1
     statements                                }
     ELSE                                      }
     DO 1 R=1.,4.,0.5                          }
          statements                           }
1    CONTINUE                                  }
ENDIF
```

DO-LOOP

DO label {I / R / D.P.} = {I / R / D.P.} , {I / R / D.P.} [, {I / R / D.P.}]

expressions

```
DO  1  R=1,15,2.1
DO  1  R=VAR,Q*SQRT(B),C*D
```

Beware of precision problems!

In DO 1 R=-0.3,-2.1,-0.3 *should have 7, may have 6 iterations*

no. of iterations =MAX(INT((m2-m1+m3)/m3),0)

*** zero trip loop possible

```
      DO    1 K = L,H,N

            H=9               <-- does not affect loop control
    1 CONTINUE
    2 I=(K-N)**2              <-- K is last count if you jump out of loop
                                  K is last count+N if drop through
```

Nesting allowed but no jump into the range of a loop is allowed from outside

CONTINUE — *do nothing*

END — *Now executable statement and may be labelled*

- *in main program acts as STOP*
- *in sub-program acts as RETURN*

```
      SUBROUTINE  XYZ
      .
      .
      GOTO 99
      .
      .
   99 END
```

STOP — *To terminate execution; not required in main program.*

RETURN — *To return control to calling program; not required before END.*

Main use now is to specify alternate returns (see later).

PAUSE — *Obsolete for batch programs.*

PROGRAM UNITS

PROGRAM — `PROGRAM name`

BLOCK DATA — `BLOCK DATA [name]`

ALL DATA statements referring to COMMON blocks must be collected in BLOCK DATA sub-programs.

May have several BLOCK DATA in one program, but only 1 unnamed.

SUBROUTINE — *SUBROUTINE name [args]*

args are - *variable name*
- *array name*
- *dummy procedure*
- *

E.g.

```
      INTRINSIC  SQRT
      .
      CALL XYZ(VAR,ARR,SQRT,*10,*20)
      .
10    .....      SUBROUTINE XYZ(A,B,FUNC,*,*)
      .          DIMENSION B(*)
      .          .
20    .....      IF (A .GT. 0) THEN
      .          B(3)=FUNC(A)
      .          RETURN 1            <---RETURNS to label 10
99    END        ELSE
                 RETURN 2            <---RETURNS to label 20
                 ENDIF
                 .
                 RETURN  I*I/4.+C
                 .  <-- expression- if result doesn't correspond to
               99 END   one of alternate returns defined (1 or 2 in
                        this example) normal RETURN is executed.
```

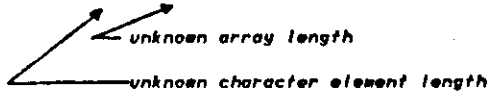
VARIABLE DIMENSIONS

Either

```
      SUBROUTINE XYZ(A,N)
      DIMENSION  A(N)
```

or

```
      SUBROUTINE  XYZ(C)
      CHARACTER *(*) C(*)
```



or

```
      SUBROUTINE  XYZ(D)
      COMMON /DIM/N,M
      DIMENSION D(M,N)
```

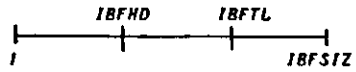
FUNCTION *statement is similar to SUBROUTINE, but no alternate returns.*

```
        LETTER = CHARF(A(3:N))
        .
        .
     99 END
        CHARACTER FUNCTION CHARF(C)
        CHARACTER*2 C
        .
        .
        CHARF = C(I:I)
        END
```

ENTRY - *Alternate entry into subprogram*

- *Arguments must agree in number and type with those in call to the entry, but can differ from those of the SUBROUTINE and other ENTRY's.*


```
      SUBROUTINE BUFS
*  BUFS IS A DUMMY ENTRY EQUATED WITH CLEAR
      LOGICAL FULL, EMPTY
      PARAMETER (IBFSIZ=100)
      CHARACTER BUF (IBFSIZ),CH
      SAVE BUF, IBFHD, IBFTL, FULL, EMPTY
      DATA IBFHD, IBFTL, FULL, EMPTY/1,1,.FALSE.,.TRUE./
*
      ENTRY CLEAR
      IBFHD = 1
      IBFTL = 1
      FULL = .FALSE.
      EMPTY =.TRUE.
      GO TO 99
*
      ENTRY PUT (CH,*)    (-- cannot be function
      IF(FULL) RETURN 1
      EMPTY=.FALSE.
      BUF(IBFTL)=CH
      IBFTL=IBFTL+1
      IF(IBFTL.GT.IBFSIZ) IBFTL=1          (-- circular buffer
      IF(IBFTL.GT.IBFHD) FULL=.TRUE.
      GO TO 99
*
      ENTRY SIZE (I,J)
      I=IBFTL-IBFHD
      IF(I.LT.0) I=I+IBFSIZ
      IF(FULL) I=IBFSIZ
      J=IBFSIZ-I        (Extract from an example of Larmouth, Salford Univ.)
   99 END
```



INTRINSIC FUNCTIONS

Some INTRINSIC functions are generic (type of function=type of argument)

and have specific names associated with them e.g.

```
A=SQRT(8)
DP=SQRT(DPDP)      } correct function supplied by compiler
COM=SQRT(COMPLX)
```

But, to pass in an argument list MUST use specific name:

```
INTRINSIC DSQRT
.
CALL SUB(DSQRT)
```

Some new functions are:

I=ICHAR(A)	I is position in collating sequence of A
A=CHAR(I)	character whose position in collating sequence is I
I=LEN(AAA)	number of characters in the character string AAA
I=INDEX(AAA,BB)	I is starting position of string BB within string AAA
LGE(AA,BB)	true if AA is lexically greater or equal to BB
LGT(AA,BB)	true if AA is lexically greater than BB
	e.g. IF(LGT(C,'H'))C=CHAR(I)
LLE(AA,BB)	true if AA is lexically less than or equal to BB
LLT(AA,BB)	true if AA is lexically less than BB
Y=ANINT(X)	Y is nearest whole number to X
I=NINT(X)	I is the nearest integer to X

New generic functions

y=MAX(x1,x2,..)	y is the maximum of the 2-500 arguments
y=MIN(x1,x2...)	y is the minimum of the 2-500 arguments
y=LOG(x)	natural LOG (to the base e)
y=LOG10(x)	LOG to base 10

Many others are consolidated, as in SQRT example.

I/O Very complete specification, with new features including

- direct access files
- internal files
- execution-time format specification
- list directed I/O
- file enquiry
- new edit descriptors

FORMATED I/O

I/O lists Input: variable, array or array element name, character substring or implied DO-list.

Output: all this plus expressions.

```
      PROGRAM IO          <-- CDC names
      OPEN (1,FILE='INPUT')
      OPEN(2,FILE='OUTPUT')
      READ(1,100)A,B,C
      WRITE(2,200)A,B*C,SQRT(A),(ODD,ODD=1.,9.,2.)
100   FORMAT(3F7.3)
200   FORMAT(1X,8F7.3)
      END
```

FORMATTED READ

```
READ (u,fmt[,IOSTAT=ios][,ERR=s1][,END=s1]) list
```

or

```
READ fmt,list
```

0 if o.k. parity error EOF check

non-0 if error

FORMATTED WRITE

```
WRITE(u,fmt[,IOSTAT=ios][,ERR=s1]) list
```

or

```
PRINT fmt, list
```

Definition of 'fmt'

a) a statement label:

```
PRINT 100,A
```

b) a character array, variable, array element or expression:

```
- PRINT '(F10.3)',A
- PRINT  F(I)//P(J),A      e.g.  F(10)='(F10'
                                 P(3) =  '.3)'
- READ (1,'(A)') FORM
```

determined by list

```
  READ (1,FORM) X,Y,Z
```

c) an integer variable to which an ASSIGN has been made

```
  ASSIGN 100 TO K
  .
  .
  PRINT K,A
  .
  .
100  FORMAT (1X,F7.3)
```

d) an asterisk - list directed I/O:

```
  WRITE (6,*) list
```


Definition of 'u':

a) unit no.:

```
READ (4,100) list
```

b) an integer expression:

```
READ (I+J-2,100) list
```

c) an asterisk:

```
READ (*,*) list
```

standard I/O file

d) a character variable, array, array element or substring identifying an internal file: (cf. ENCODE/DECODE)

Ex.1

```
      DIMENSION IN(78)
      CHARACTER CARD*78, FM(3)*6
      DATA FM/'(78I1)','(39I2)',
     *        '(26I3)'/
      READ(5,'(I1,A)',END=100)KEY,CARD
      N= MAX(1,MIN(KEY,3))
      READ(CARD,FM(N)) (IN(I),I=1,78/N)
      .
      .
*     NO DATA ENCOUNTERED
  100 .....
```

Ex.2

```
      CHARACTER*8 BIRD(3),A*1,B,C
      A='Z'
      B='ABCDE'
      C='12345678'
      WRITE (BIRD,'(A1/A5/A8)')A,B,C
```

```
BIRD(1)    'Z'
BIRD(2)    'ABCDE'
BIRD(3)    '12345678'
```

each array element corresponds to a new record

I/O CONTINUED

Edit Descriptors

repetitive:

Ew.d	Ew.dEe (o/p only)	E10.2E3
Fw.d		-9.99E-57
Dw.d		
Gw.d	Gw.dEe (o/p only)	
Iw	Iw.m -"-	I5.3
Lw		^^^099
Aw	A (list directed)	

Non-repetitive:

BN	*(default)*	BN,I3 --) 14
	1^4	
BZ		BZ,I3 --) 104
SP	SP,I3 --) +99	
SS	SS,I3 --) 99	
S	*(machine default,= SS on IBM and CDC)*	
nX		
Tn	READ (*,'(T3,I3,TL3,I1,I2)')I,J,K	
TRn	^^^987	
TLn	→ I=987,J=9,K=87	
nH	*IIHI MUST COUNT*	
'	'THIS IS EASIER'	
:	PRINT '(''NO OF TRACKS='',I4:)',(TBANK(I),I=1,5)	
/		
kP		

UNFORMATTED I/O

Form of READ and WRITE statements already described, but without format specification.

DIRECT ACCESS FILES

Files whose records can be read, written or rewritten without regard for the concept of position.

```
      DIMENSION A(0:119), B(101:220)
      OPEN (2,FILE='MYFILE',ACCESS='DIRECT',RECL=120)
      .
      .
      WRITE (2,REC=14)B
      .
      .
      READ (2,REC=14)A
            .
            .
            DO 1 I=0,119
    1       A(I)= 2.**I
            .
            .
      WRITE (2,REC=14)A
```

Can be formatted by adding FORM='FORMATTED' *to OPEN statement.*

Very simple

Very powerful

I/O STATUS STATEMENTS

OPEN	OPEN(u,	unit no.
	IOSTAT=ios,	0 if no opening error, otherwise non-0
	ERR=sl,	control to sl if error on opening
	FILE=fln,	file name of file
	STATUS=st,	'OLD'
		'NEW' for new association with u
		'SCRATCH'
		'UNKNOWN' default
	ACCESS=acc,	'SEQUENTIAL' default
		'DIRECT'
	FORM=fm,	'FORMATTED' default for sequential
		'UNFORMATTED' default for direct
	RECL=rl,	Obligatory for direct
	BLANK=bl)	'NULL' default
		'ZERO' cf. BN,BZ edit descriptors

CLOSE CLOSE (u[,IOSTAT=ios][,ERR=sl][,STATUS=st])

st= 'KEEP' default

'DELETE' default for SCRATCH files

FILE CONTROL STATEMENTS

BACKSPACE u

REWIND u

ENDFILE u

INQUIRE

INQUIRE ({ u / FILE=fln } ,	unit no. / file name
IOSTAT=ios,	error no., 0 if no error
ERR=sl,	error exit
EXIST=ex,	.TRUE. if exists, otherwise .FALSE.
OPENED=op,	.TRUE. if connection between 'u' and 'fln'

IF ex=.TRUE.

NAMED=nmd,	.TRUE. if named
NAME=nam,	character variable contains name of file
SEQUENTIAL=seq,	'YES','NO', or 'UNKNOWN' depending whether <u>can</u> be opened for seq.
DIRECT=dir,	-"- for direct
FORMATTED=fmt,	'YES','NO', or 'UNKNOWN' depending whether <u>can</u> be opened for formatted i/o
UNFORMATTED=unf,	-"- for unformatted i/o

IF op=.TRUE.

NUMBER=num,	unit no. associated with file
ACCESS=acc,	'SEQUENTIAL' or 'DIRECT'
FORM=frm,	'FORMATTED' or 'UNFORMATTED'

IF acc= 'DIRECT'

RECL=rec,	record length in characters (formatted) or in words (unformatted)
NEXTREC=nr,	no. of last record read or written +1

IF frm= 'FORMATTED'

BLANK=bl,	'NULL' or 'ZERO'

Have assumed INQUIRE by file name, similar for INQUIRE by unit no.

EXAMPLES

Ex.1

```
      PROGRAM PASCAL
*
*   TO PRINT A PASCAL TRIANGLE WITH NR ROWS
*
      PARAMETER (NR=15)
      DIMENSION LROW(NR)
      DATA LROW/NR*1/
*
*   INITIAL PRINT
*
      PRINT '('''1 PASCAL TRIANGLE''//1X,I5/2I5)',
     *LROW(NR),LROW(NR-1),LROW(NR)
*
*   FORM COEFFICIENTS
*
      DO 50 J=NR-1,2,-1
         DO 40 K=J,NR-1
40       LROW(K)=LROW(K)+LROW(K+1)
         PRINT '(1X,(I5))',(LROW(M),M=J-1,NR)
50    CONTINUE
      END
```


Ex.2

```
      PROGRAM BLOCK
*
*   TO CALCULATE C DEPENDING ON K
*
      PARAMETER (M=5)
      DIMENSION A(M),B(M),C(M)
*
*   READ DATA FROM STANDARD INPUT UNIT
*
    2 READ (*,*,END=100) K,A,B
      IF(K.EQ.1) THEN
         DO 5 I=1,M
    5    C(I)=A(I)**2+B(I)**2
         ELSEIF (K.EQ.2) THEN
         DO 10 I=1,M
   10    C(I)=A(I)*B(I)
         ELSE
         DO 15 I=1,M
   15    C(I)=0.
      ENDIF
*
*   PRINT RESULT
*
         PRINT '(1X,I5/(5E10.3))',K,A,B,C
         GO TO 2
  100    END
```

Ex.3

```
      PROGRAM ASK
      LOGICAL E,O
      CHARACTER*10 N,A,S,F*11
      DIMENSION BUF(50)
      DATA BUF/50*1./
*
*   OPEN FILE
      OPEN(2,ERR=99,FILE='N123',STATUS='NEW',
     *   ACCESS='DIRECT',RECL=100)
      K=1
      DO 15 I=1,*1,10
        WRITE(2,REC=K,ERR=99)(BUF(J),J=I,I+9)
   15 K=K+1
*
   FILE ENQUIRY
      INQUIRE(2,ERR=99,EXIST=E,OPENED=O,NAME=N,
     * ACCESS=A,SEQUENTIAL=S,FORM=F,RECL=L,NEXTREC=M)
      PRINT *,E,O,N,A,S,F,L,M
```

T — *E*
T — *O*
N123 — *N*
6 — *L*
100 — *M*
UNFORMATTED — *F*
NO — *S*
DIRECT — *A*

```
      GOTO 999
   99 PRINT *,'FATAL ERROR'
  999 END
```


Ex.4

```
      SUBROUTINE STAR (C,*)
*
*   TO DEMONSTRATE (OVER)USE OF *
*
      CHARACTER *(*) C(*)
      READ (*,*) K
      IF (LLE(C(2)(:1),'*') THEN
            RETURN 1
            ELSE
            CALL ASTER (K,*10,*99)
            K=4**K+K**2-
     *           K**K
       ENDIF
   10 PRINT *,K
   99 END
```

Ex.5

```
      SUBROUTINE SUB(A)
      DIMENSION A(*)
      .
      .
      ENTRY BANG(X,K)
      .
      .
      X= A(K)      illegal: either A, or X and K, are undefined
      .
      .
      END
```

BACKWARDS COMPATIBILITY OF ANSI CODE

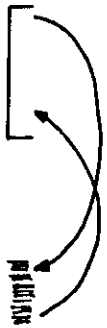
Principal Differences:

- *cols. 1-5 of continuation cards must be blank*
- *Hollerith constants and data are allowed*
- *value of subscripts may not exceed bounds:*

```
DIMENSION A(10,5)
Y=A(11,1)
```

- *in EQUIVALENCE, the number of array subscripts, if given, must agree with the declaration:*

```
DIMENSION B(2,3,4),C(4,8)
EQUIVALENCE (B(23),C)
```



- *no 'extended-range' Do-loop:*

- *Intrinsic function passed in an argument must be declared as INTRINSIC not EXTERNAL*

ELEMENTARY USE OF FORTRAN 77

IBM VM/CMS

Use **VFORT**

See: VM/CMS Tutorial DD/US/32 Chapter 9.

Further details: *'Effective FORTRAN 77', M. Metcalf, Oxford University Press, 1985.*

Optimization and Portability Considerations: *see 'FORTRAN Optimization', M. Metcalf, Academic Press, October 1985.*